IRIA

CENTRE DE ROCQUENCOURT

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P.105
78153 Le Chesnay Cedex
France
Tel.(3) 954 90 20

Rapports de Recherche

N° 306

NUMERICAL OPTIMIZATION OF ENERGY-PRODUCTION SYSTEMS

Maria Cristina BANCORA-IMBERT
Roberto GONZALEZ
Jean-Claude MIELLOU
Edmundo ROFMAN

Mai 1984

"NUMERICAL OPTIMIZATION OF ENERGY-PRODUCTION SYSTEMS"

Maria Cristina BANCORA-IMBERT

Roberto GONZALEZ

Jean Claude MIELLOU

Edmundo ROFMAN

NUMERICAL OPTIMIZATION OF ENERGY-PRODUCTION SYSTEMS

M.C. IMBERT[1], R. GONZALEZ[2], J.C. MIELLOU[3] and E. ROFMAN[1]

ABSTRACT

The paper proposes new methods for solving the sequence of non linear fixed-point problems related to the optimization of energy production systems (see [9]) and presents results obtained in treatment of more complex systems. As a consequence of the significant reduction of the number of operations, some sytems can be optimized in real time and more complex ones advantageously studied.

RESUME

De nouvelles méthodes sont proposés pour resoudre la suite des problèmes de point fixe associé à l'optimisation des systèmes de production d'énergie (cfr. [9]). Grâce à la réduction du nombre des opérations quelques systèmes peuvent être optimisés en temps réel. Des résultats numériques sont présentés.

(1) INRIA

(2) CONICET and UNIV. NAC. ROSARIO, BV. ORONO 2612, 2000 ROSARIO, ARGENTINE.

(3) LABORATOIRE D'ANALYSE NUMERIQUE, UNIV. BESANCON, ROUTE DE GRAY 25030 BESANCON CEDEX, FRANCE.

INTRODUCTION

The aim of the paper is to show the impact of several recent results in the resolution of the problem presented by Gonzalez and Rofman [9] . In that paper a new approximation procedure for computing the optimal cost of deterministic control problem was given and a first test was made on an energy production system of two thermopower plants and a dam.

In §1 and §2 we recall the results that are necessary to discuss a more general short-run model. This model is presented in §3; μ hidropower and ν thermopower plants are now considered in the production system. The quasi-variational inequalities to be satisfied by the value function are given at the end of this chapter.

In §4 it is shown that we deal with fixed-point problems belonging to a class of problems for which special relaxation-type alorithms can be used. In particular a "mono-iteration relaxation" algorithm is proposed.

In §5 the fixed-points problems are considered as dynamic programming problems on a graph. This approach is possible thanks to the form of the number of states n . In fact, in our problem $n=2^{\nu}$. Comparative exemples show the advantadge of this approach. Particular efforts were necessary in the computing programming task to allow the treatment of more complicated systems. Automatic trajectory simulation routines were also written.

At the end of the paper two problems presented by E.D.F. are solved.

§ 1 THE ORIGINAL PROBLEM AND ITS EQUIVALENT FORMULATION

The system satisfies in absence of impulse controls the differential equation

$$(1,1)\qquad \left|\begin{array}{ll} \frac{dy}{ds} = f(y,u,s) & x \in \Omega \subset \mathbb{R}^n \\ y(t) = x & t \in [o,T] \end{array}\right.$$

where $u(.)$ is a measurable function of the time, with values in a compact set $U \subset \mathbb{R}^m$.

In a finite set of times $\theta_\nu (\nu = 1,2,\ldots\mu)$ impulses $z(\theta_\nu) \in Z$ are applied ; the trajectory jumps are

$$(1,2)\qquad y(\theta_\nu^+) = y(\theta_\nu^-) + g(y(\theta_\nu^-),\ z(\theta_\nu),\theta_\nu)$$

Z is a compact set of $\mathbb{R}^n$.

We denote by $(u(.),z(.),\tau)$ a control strategy with the stopping-time $\tau \in [o,T[$.

The cost associated with each strategy is

$$(1,3)\qquad J(x,t;u(.),z(.),\tau) = \int_t^{\tau\wedge T} e^{-\alpha(s-t)} \ell(y(s),u(s),s)\, ds + \sum_\nu q(y(\theta_\nu^-), z(\theta_\nu),\theta_\nu) e^{-\alpha(\theta_\nu - t)} + e^{\alpha(\tau-t)} \phi(y(\tau),\tau) \chi_{[t,T[}$$

$\chi_{[t,T[}(.)$ characteristic function of the interval $[t,T[$.

The optimal cost function is

$$(1,4)\qquad V(x,t) = \inf \{ J(x,t;u(.),z(.),\tau) : u(.),\ z(.),\tau \} \ ,\ \forall (x,t) \in Q$$

$$(1,5)\qquad Q = \Omega \times [0,T]$$

In the following we will suppose

i) f, ℓ, $\emptyset$, g, q are continuous and bounded functions ; they are lipschitzean functions in (x,t).

ii) $\emptyset(x,T) \geq 0$, $\forall\, x$

iii) $q(x,z,t) \geq q_0 > 0 \qquad \forall (x,t) \in Q, \quad \forall\ Z$

iv) $\forall\ t$, $y(t) \in \Omega$ independently of the strategy.

We can give the following characterization of $V(x,t)$

Theorem 1.1 :

$V(x,t)$ is the maximum element of the set W, with

$W = \{w(x,t) \to \mathbb{R} \,/\, (1.6) - (1.10)\}$

(1.6) $w(x,t)$ lipschitzean function in (x,t);

(1.7)
$$\frac{\partial w(x,t)}{\partial t} + \min_{u \in U} \left[\frac{\partial w(x,t)}{\partial x} \; f(x,u,t) + \ell(x,u,t) - \alpha\, w(x,t) \right] \geq 0$$

a.e. $(x,t) \in Q$;

(1.8)
$$w(x,t) \leq \min_{z \in Z} \left(q(x,z,t) + w\,(x+g(x,z,t),t) \right)$$

$\forall\ (x,t) \in Q$;

(1.9) $w(x,t) \leq \emptyset(x,t)$, $\forall\ (x,t) \in Q$

(1.10) $w(x,T) \leq 0$, $\forall\ x \in \Omega$.

The proof follows the method used in [9], p. 29 .

§ 2 THE APPROXIMATION PROCEDURE FOR THE OPTIMAL COST

2.1 The discretized problem (P_h)

a) The set Q is approximated with a triangulation Q^h, union of simplices of vertices (x_p,t_p) ; $p = 0,N_x$; $q = 0, N_T$, $t_q = q\delta$, $\delta = \frac{T}{N_T}$

b) In the set of linear finite elements w^h defined in Q^h we consider the set W^h:

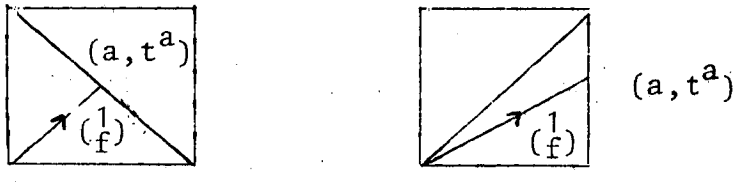
$$W^h = \{w^h : Q^h \longrightarrow \mathbb{R} \,/\, (2.1),\ (2.3),\ (2.4),(2.5)\}$$

$$(2.1)\qquad \frac{\partial w^h}{\partial t}(x_p,t_q;u) + \frac{\partial w^h}{\partial x_f}(x_p,t_q,u)\,\|f(x_p,u,t_q)\| + \ell(x_p,u,t_q) - \alpha\, w^h(x_p,t_q) \geq 0$$

where $\frac{\partial w^h}{\partial x_f}$ is the derivative of w^h in the direction of the vector f and

$$\frac{\partial w^h}{\partial t}(x_p,t_q;\,u) + \frac{\partial w^h}{\partial x_f}(x_p,t_q\,;\,u)\ \|f(x_p,u,t_q)\|$$

is the product of the derivative of w^h in the direction of the vector $(1,f'(x_p,u,t_q))' \in \mathbb{R}^{n+1}$ by the norm of such vector ; for example, in the situation depicted in the following figure



the expression is equal to

$$(2.2)\qquad \frac{w^h(a,t^a) - w^h(x_p,t_q)}{\Delta}$$

with $\Delta = t^a - t_q$;

$$(2,3)\qquad w^h(x_p,t_q) \le q(x_p,z,t_q) + w^h(x_p+g(x_p,z,t_q),t_q)$$

$$\forall\, z \in Z^h,\ \forall x_p,\ p = 0,\ N\ ;\ \forall\, t_q,\ q = 0, N_T-1$$

$$(2,4)\qquad w^h(x_p,t_q) \le \emptyset\ (x_p,t_q)\quad \forall p = 0, N_x\ ,\ \forall q = 0, N_T-1$$

$$(2,5)\qquad w^h(x_p,t_{N_T}) \le 0\ ,\quad \forall\ p = 0,\ N_x$$

Remark : The sets U^h, Z^h must satisfy some "consistency conditions" as in ([9] ;2.1,d).

c) We introduce the following partial order "≤"

$$(2,6)\qquad w^h \le \tilde{w}^h \iff w^h(x_p,t_q) \le \tilde{w}^h(x_p,t_q)\ ,\ \forall\ p = 0,\ M_x\ \ ;\ q = 0,\ M_T$$

and we pose the discretized problem :

(P_h) : Find the maximum element $\bar{w}^h$ of the set W^h with respect to the partial orden " ≤ ".

2.2 The solution of (P_h) and its properties

(2.1) and (2.3) will be transformed into equivalent and more useful relations.

Taking into account that in (2.2), the points (a,t^a) are, in general, interior points of aces (or edges) of some simplex, we will express these points as convex combinations of the vertices of the faces to which they belong and by the affinity of w^h, (2.1) becomes

$$(2,7)\qquad w^h(x_p,t_q) \le \min_{x \in U^h} \frac{1}{(1+\alpha\Delta)} \{\ \frac{\Delta}{\delta} \sum_j \lambda_j(x_p,t_q,u)\ w^h(x_j,t_{q+1}) + (1-\frac{\Delta}{\delta}) \sum_j \hat{\lambda}_j(x_p,t_q,u)$$

$$w_h(x_j,t_q) + \Delta \ell(x_p,u,t_q)\}$$

in which

$$(2,8)\qquad \delta = t_{q+1}-t_q$$

$$(2,9)\qquad a = \frac{\Delta}{\delta}\ (\sum_j \lambda_j.x_j) + (1-\frac{\Delta}{\delta}) \sum \hat{\lambda}_j.x_j)$$

$$(2.10)\qquad \begin{cases} \sum_j \lambda_j(x_p,t_q,u) = 1 \\ \sum_j \lambda_j(x_p,t_q,u) = 1 \end{cases}$$

In the same way we put

$$(2.11)\qquad x_p + g(x_p,z,t_q) = \sum_j \lambda'_j(x_p,t_q,z)x_j$$

and (2.3) is rewritten in the equivalent form

$$(2.12)\qquad \begin{cases} w^h(x_p,t_q) \le \min_{z\in Z^h}\{q(x_p,z,t_q) + \Sigma\lambda'_j(x_p,t_q,z)w^h(x_j,t_p)\} \\ \forall\ p=0,N_X\ ;\ q=0,N_T-1 \end{cases}$$

We will use (2.7) and (2.12) to define the operator M, (w^h denotes a linear finite element in Q^h) :

$$(2.13)\qquad \begin{cases} \text{if } q = N_T \quad (Mw^h)(x_p,t_q) = 0 \\ \text{if } q = 0,\dots,N_T-1 \quad (Mw^h)(x_p,t_q) = \min\{\emptyset(x_p,t_q), \\ \qquad \min_{z\in Z^h}(q(x_p,z,t_q) + \sum_j \lambda'_j(x_p,t_q,z)\, w^h(x_j,t_q)), \\ \qquad \min_{u\in U^h} \frac{1}{1+\alpha\Delta}\left[\frac{\Delta}{\delta}\sum_j \lambda_j(x_p,t_q,z)w^h(x_j,t_{q+1}) + (1-\frac{\Delta}{\delta})\right. \\ \qquad \left.\sum_j \hat{\lambda}_j(x_p,t_q,u)w^h(x_j,t_q) + \ell(x_p,u,t_q)\right]\}. \end{cases}$$

We define Mw^h at arbitrary points in Q^h by linear interpolation of the values given by (2.13) at the vertices of the triangulation.

Some properties of Mw^h which follow immediately are :

$$(2.14)\qquad w^h \ge \hat{w}^h \Longrightarrow Mw^h \ge M\hat{w}^h$$

$$(2.15)\qquad w^h \in W^h \Longleftrightarrow w^h \le Mw^h$$

Remark : (2.15) give us a characterization of W^h.

Finally the most important property is given by

Theorem 2.1 (see [9])

There exists $\bar{w}^h$, maximum element of W^h; furthermore $\bar{w}^h$ is characterized by the condition $\bar{w}^h = M\bar{w}^h$, i.e.

$$\bar{w}^h = M\bar{w}^h \iff \bar{w}^h \geq w^h, \ \forall w^h \in W^h \tag{2.16}$$

2.3 Algorithm to compute $\bar{w}^h$

To compute $\bar{w}^h$, we can use the following algorithm A1. This algorithm is an improvement of algorithm (I.2.34)[9] because it takes advantage of the particular structure of non-stationnary problems (we use backward solutions) and it computes and approximate solution $\bar{w}_\varepsilon^h$ in a finite number of steps.

Algorithm A1 :

step 0 : choose $\varepsilon > 0$, $\tilde{w}^h \in W^h$ set $w_R^h = \tilde{w}^h$, $\hat{w}^h = \tilde{w}^h$, $w^h = \tilde{w}^h$, $q = N_T$

step 1 : set $p = 0$

step 2 : set $w^h(x_p, t_q) = (Mw^h)(x_p, t_q)$

step 3 : if $p = N_x$, go to step 5 ; else, go to step 4.

step 4 : set $\hat{w}^h(x_p, t_q) = w^h(x_p, t_q)$, and go to step 2 .

step 5 : if $w^h(x_p, t_q) \leq w_R^h(x_p, t_q) + \varepsilon$ for every $p = 0, N_x$ go to step 6; else go to step 7.

step 6 : if $q = 0$, set $\bar{w}_\varepsilon^h = w^h$ and stop ; else, set $q = q-1$ and go to step 1.

step 7 : set $w_R^h(x_p, t_q) = w^h(x_p, t_q)$ for every $p = 0, N_x$ and go to step 1.

In fact, it is possible to show the following theorem :

Theorem 2.2 :

The algorithm A1 stops after a finite number of iterations at the element $\bar{w}_\varepsilon^h$, having the following properties :

a) $\bar{w}_\varepsilon^h \in W^h$, $\forall \varepsilon > 0$

b) $\varepsilon \leq \varepsilon' \Longrightarrow \bar{w}_\varepsilon^h \geq \bar{w}_{\varepsilon'}^h$

c) $\lim_{\varepsilon \to 0} \bar{w}_\varepsilon^h = \bar{w}^h$

2.4 The convergence of the approximate solutions

It is possible to prove a theorem similar to Theorem I.3.2 [9]

Theorem 2.3 :

The approximate solution $\bar{w}^h$ converges uniformly to V(x,t) i.e.

$$\lim_{\|h\| \to 0} \max_{(x,t) \in Q^h} |\bar{w}^h(x,t) - V(x,t)| = 0$$

§ 3 THE OPTIMAL CONTROL OF AN ENERGY-PRODUCTION SYSTEM

3.1 Modelisation of the problem (short-run model)

The energy production system consists of ν thermopower plants ($P_1, P_2, \ldots, P_\nu$ being their level of production) and μ hydroplants ($y_1, \ldots y_\mu$: hydropower stock ; $\Pi_1, .. \Pi_\mu$: hydropower production). D is the demand of electricity and we denote by $P_{\nu+1}$ the production of an additional source, which is available if it is required.

$$D = \sum_{r=1}^{\nu} P_r + \sum_{\ell=1}^{\mu} \Pi_\ell + P_{\nu+1} \tag{3.1}$$

The cost of the operation is given by

$$J = \int_0^T \left[\sum_{r=1}^{\nu} c_r P_r(s) + \sum_{\ell=1}^{\mu} c_{h\ell}\,(y_\ell(s))\Pi_\ell(s) + c_{\nu+1}\, P_{\nu+1}\,(s) \right] ds + \sum_{r=1}^{\nu} n_r\, \bar{k}_r \tag{3.2}$$

n_r is the number of starts of the r thermopower plant, in the interval $[0,T]$.

$\bar{k}_r$ is the cost of each start up.

We suppose c_r, $j=1,..,\nu$ constants and $c_{h\ell}$ are shadow prices obtained after a long-run optimization (about one year). In our problem, we will consider $[0,T]$ one day or one week. (see [10]) .

We will suppose that there are not delays between the start up of termal plant and the instant in which it begins to produce energy. The methodology to be used here can be easily modified to take into account these delays (see [8])

In this form, the system will be modeled by its internal state (a discrete variable $E = 1,2,3,\ldots, 2^\nu$ showing which thermopower plants are operating) and the continuous variables y_ℓ whose evolution equations are

$$\frac{dy}{dt} = A(s) - \Pi(s) \tag{3.3}$$

$$y=(y_1 \ldots, y_\mu) \quad \Pi=(\Pi_1, .., \Pi_\mu)$$
$$A=(A_1 \ldots A_\mu)$$
$$0 \le y_\ell \le y_{\ell,\max} \qquad \ell = 1, \ldots, \mu$$

Where $A_\ell(s)$ is the imput of water in the ℓ - plant.

Our aim is to obtain the control strategy giving the minimum of J. The optimal strategy is a decision concerning when the power plants must operate and at what level of production. We look for optimal feed-back policies acting on the instantaneous state (E(s), y(s)) of the system.

3.2 Optimal feed-back policies

Let us consider as parameters the initial hydropower stock $x = (x_1, .. x_\mu)$ and the initial time t of the system and let us introduce the optimal cost functions $V_i(x,t)$, $i=1,..,2^\nu$, $(x,t) \in Q = \prod_{\ell=1}^{\mu} ([o, y_{\ell,\max}] \times [o,T]$:

$$(3.4) \quad V_i(x,t) = \inf_{P_1(.),\ldots,P_\nu(.),\Pi_1(.),\ldots,\Pi_\mu(.)} J(x,i,t;P_1(.),\ldots,P_\nu(.),\Pi_1(.)..\Pi_\mu(.))$$

with J given by (3.2) related to the initial data $(E(t),y(t))=(i,x)$.

In the following we shall note $P(.)=(P_1(.)\ldots P_\nu(.))$

From $V_i(x,t)$ it will be possible to define the optimal feed-back policies (see [7]). So, our problem is to compute $V_i(x,t)$.

We recall the following :

3.3 Quasi-variational inequalities (QVI) associated with the control problem and characterization of V_i .

It is possible to show (see [7]) that the V_i's are differentiable in a.e $(x,t) \in Q$. Furthermore they verify a.e. in Q (see [7],[10]) the system of Q.V.I. ($i = 1,\ldots,2^\nu$) :

$$(3.5) \quad \frac{\partial V_i}{\partial t} + \min_{(P,\Pi) \in \Gamma_i(x)} \Big(\sum_{\ell=1}^{\mu} \frac{\partial V_i}{\partial x_\ell} (A_\ell - \Pi_\ell) + \sum_{r=1}^{\nu} c_r P_r + \sum_{\ell=1}^{\mu} c_{h\ell}(x)\Pi_\ell + c_{\nu+1}(D - \sum_{1}^{\nu} P_r - \sum_{1}^{\mu} \Pi_\ell)^+ \Big) \geq 0 \ ;$$

(3.6) $$V_i(x,t) \leq V_j(x,t) + k_j^i \ , \quad \forall j \neq i \ ;$$

(3.7) $$V_i(x,T) = 0 \ ;$$

(3.8) For a.e. $(x,t) \in Q$ one at least of (3.5) or (3.6) is an equality,

with $\Gamma_i(x)$ the set of admissible levels of production related to the state i and the initial stock x; k_i^j the cost for passing from state i to state j $(j=1,2,\ldots,2^\nu)$.

The following characterization of $V_i(x,t)$ will allow us to compute it using the approximation procedure introduced in § 2

$V_i(x,t)$ is the maximum element of the set W_i :

(3.9) $$W_i = \{w_i \in H^{1,\infty}(Q) / w_i \text{ verifies (3.5), (3.6), (3.7)}\}$$

$$\text{i.e. } w_i(x,t) \leq V_i(x,t) \ , \quad \forall (x,t) \in Q, \quad \forall w_i \in W_i$$

§ 4 - A FIXED POINT PROBLEM ; A LINK BETWEEN RELAXATION METHOD AND MOORE-DIJSKTRA ALGORITHM.

4.1. Setting the problem ; background about relaxation method.

After discretization the system (3.5) to (3.8) gives rise to the problem (2.13), which leads at each time step and each space step to the resolution of a fixed point problem that we may describe in the following form :

With the hypothesis :

(4.1) $\left\{ \begin{array}{l} \text{Let } \phi = \{\ldots,\phi_i,\ldots\}^T \in R^n \text{ ; the components } \phi_i \text{ of } \phi \text{ being all} \\ \\ \text{non negatives.} \end{array} \right.$

(4.2) $\left\{ \begin{array}{l} \text{Let K be a square } \{n,n\} \text{ matrix whose coefficients } k_{i,j} \text{ belong} \\ \text{to R. All these coefficients are supposed to be non negative.} \end{array} \right.$

We associate to ϕ and K verifying (4.1) - (4.2), the non linear application F , with domain equal to the cone C of vectors of R^n with all components non negative, defined by :

We denote by $a \wedge b$ the minimum of $a, b \in R$.

(4.3) $\left\{ \begin{array}{l} \text{Let } w \in C, \text{ then } F(w) \in C \text{ is such that if } F(w) = \{\ldots,F_i(w),\ldots\}^T, \\ \\ \forall i \in \{1,\ldots,n\} \ , \ F_i(w) = \left(\underset{j \neq i}{\wedge} (k_{i,j} + w_j) \right) \wedge \phi_i \end{array} \right.$

and we consider the fixed point problem :

(4.4) Find $u \in C$ such that : $u = F(u)$

Remarks and comments :

i) we are interested with the calculation of a maximal solution of problem (4.4).

ii) in our problem the coefficients of K, and the components of ϕ are taken in R, but there are numerous problems, especially those connected with graph theory, which lead to fixed point equations of the kind (4.3) - (4.4), including situations in which in (4.1) to (4.4) R must be replace by : $R \cup \{+\infty\}$.

A typical such situation is consituted by the problem of the determination of all shortest routes from a fixed origin in a graph with non negative arc length, which in our framework corresponds, for example, to cases where $\phi_1 = 0$, and $\phi_i = +\infty$, for $i > 1$.

In fact we give below a presentation of the classical MOORE-DIJSKTRA algorithm without any reference to graph theory (cf. [5] chap. 2 and 3 for the graph theoretic point of view about this subject).

Our own point of view provides conversely a simple interpretation of MOORE-DIJKSTRA algorithm in terms of a particular relaxation method.

Background about relaxation method :

Let $w^o = \{\ldots, w_j^o, \ldots\}^T$ be an initial data, we consider the algorithm :

$$(4.5) \quad \begin{cases} w^{p+1} = F(w^p) & \forall p \in N. \\ \text{(successive approximation method).} \end{cases}$$

$$(4.6) \quad \begin{cases} \tilde{w}_i^{p+1} = F_i(\tilde{w}_i^{p+1}, \ldots, \tilde{w}_{i-1}^{p+1}, \tilde{w}_{i+1}^{p}, \ldots, \tilde{w}_n^{p}), \\ \forall i \in \{1, \ldots, n\} \ , \ \forall p \in N. \\ \text{(non linear Gauss-Seidel's method).} \end{cases}$$

It is easy to verify that in the case of (4.3) :

$$(4.7) \qquad w \leq w' \Longrightarrow F(w) \leq F(w')$$

If moreover, w^o is a super-solution (sometimes also called "post-fixed point"), of equation (4.4), that is to say if $F(w^o) \leq w^o$, then it is classical that the sequences $\{w_p\}$ and $\{\tilde{w}^p\}$ are monotonically decreasing and convergent. For this kind of result we refer to KANTOROVICH Lemma : cf. Chap. 13 of [14]).

It is also classical to consider relaxation method in a rather general form :

$$(4.8) \quad \left\{ \begin{array}{l} \forall p \in N \quad \text{let } J_p \subset \{1,\ldots,n\} \text{ be such that } J_p \neq \emptyset \quad \text{and} \\ \forall i \in \{1,\ldots,n\} \ \{p \in N,\ i \in J_p\} \text{ be infinite,} \\ \text{with initial data } \bar{w}^o = w^o. \\ \bar{w}_i^{p+1} = F_i(\ldots,\bar{w}_j^p,\ldots) \qquad \text{if } i \in J_p \\ \bar{w}_i^{p+1} = \bar{w}_i^p \qquad \text{if } i \notin J_p \end{array} \right.$$

with, for J_p, the choice :

(1) $\forall p \in N : J_p = \{1,\ldots,n\}$ we get (4.5).

(2) $\forall p \in N : J_p = \{1 + p \bmod (n)\}$ we get (4.6).

The monotonicity properties quoted above for (4.5) and (4.6), are also true for (4.8) (upon this topic studied in more general cases -asynchronous iterations-we refer to [3] , [4] , [11], [12], [13].)

4.2. A presentation of the Moore-Dijsktra algorithm as a "mono-iteration" relaxation"algorithm.

We call here "iteration" the computational work associated to one evaluation of all components F_i ($i \in \{1,\ldots,n\}$) of F (for two distinct components F_i and F_j the respective vectorial arguments being possibly

distinct, like it is the case for Gauss-Seidel's method) ; for example, the passage from w^p to w^{p+1} or from $\tilde{w}^p$ to $\tilde{w}^{p+1}$ respectively in (4.5), (4.6) are such iterations.

It is important to avoid all possible confusion between such iterations and the loops of the program below , where one loop is associated with the evaluation of only one component F_i (or with someones when card $J_m > 1$; in (4.8) such a loop correspond to the passage of $\bar{w}^p$ to $\bar{w}^{p+1}$ and we remark that in the case of algorithm (4.5) and only in this case : loop equal iteration).

"Mono-iteration" relaxation algorithm : form 1 :

Let us consider the following program :

Datas : Integer n ; vector ϕ ; matrix K

"Initializations" $J = \{1,\ldots,n\}$
while $J \neq \emptyset$ do

"loop" begin ;

Let J_m be included in

$$\{i_m \in J / \phi_{i_m} = \bigwedge_{j \in J} \phi_j\} \text{ and } J_m \neq \emptyset$$

$$J = J \ominus J_m \; ; \; \forall i \in J_m \quad u_i = \phi_i \; ;$$

$$\forall i \in J \quad \phi_i = \left(\bigwedge_{j \in J_m} (k_{i,j} + u_j) \right) \wedge \phi_i ;$$

end "loop ;
end ;

Proposition 4.1 : The hypotheses (4.1) to (4.3) being satisfied the above algorithm computes the components u_i of a maximal solution u of problem (4.4) according to their increasing order, using at most n loops of the corresponding program.

Proof : Let ℓ be the index associated to the ℓ-th execution of the loop of the algorithm. At the beginning of the ℓ-th loop, let therefore $J^{\ell-1}, J_m^{\ell-1}$ be the respective states of J_m, J ; and let $\phi_i^{\ell-1}$ be the state of the i-th component of ϕ.

When $\ell=0$, we have :

$$J_m^o = \emptyset \quad ; \quad J^o = \{1,\ldots,n\};$$

$$\forall i \in J_m^1 \ , \ u_i = \phi_i^o = \phi_i \ ; \ J^1 = J^o \ominus J_m^1 \ ;$$

$$\forall i \in J^1 \ , \phi_i^1 = \Big(\bigwedge_{j \in J_m^1} (k_{i,j} + u_j) \Big) \wedge \phi_i$$

Let :

$$\forall \ell > 1 \ / \ J^\ell \neq \emptyset \quad \mathcal{J}^\ell = \bigcup_{r \leq \ell} J_m^r$$

We consider now the two following induction hypotheses :

(4.9)
$$\begin{cases} \text{Let } \ell', \ell'' \text{ be integers with } \ell' < \ell'' \leq \ell, \\ \text{then } \forall i \in \mathcal{J}^\ell \ , \ u_i \text{ is well defined, and we have :} \\ \forall i' \in J_m^{\ell'} \ , \ \forall i'' \in J_m^{\ell''} \quad u_{i'} \leq u_{i''} \end{cases}$$

and :

(4.10) $$\forall i \in J^{\ell-1}, \phi_i^{\ell-1} = \Big(\bigwedge_{j \in \mathcal{J}^{\ell-1}} (k_{i,j} + u_j) \Big) \wedge \phi_i \ ;$$

by the definition of J_m^ℓ, J^ℓ, we have :

$$\forall i \in J_m^\ell \quad u_i = \phi_i^\ell = \bigwedge_{j \in J^{\ell-1}} \phi_j^{\ell-1}$$

$$J^\ell = J^{\ell-1} \ominus J_m^\ell \ ; \text{ therefore :}$$

(4.11) $$\forall i \in J^\ell \ , \ \forall j \in J_m^\ell \ , \ \phi_i^{\ell-1} \geq \phi_j^{\ell-1} \ .$$

By the hypothesis (4.2) K is non negative, which especially implies :

$$\forall i \in J^\ell, \ \forall j' \in J_m^\ell \quad k_{i,j'} + \phi_{j'}^{\ell-1} \geq \phi_{j'}^{\ell-1}$$

which, with (4.11), implies :

$$(4.12) \qquad \forall i \in J^{\ell} \ , \ \forall j \in J_m^{\ell} \ , \ \phi_i^{\ell} = \left(\bigwedge_{j' \in J_m^{\ell}} (k_{i,j'} + \phi_{j'}^{\ell-1}) \right) \wedge \phi_i^{\ell-1} \geq \phi_j^{\ell-1} \ .$$

As $J_m^{\ell+1} \subset J^{\ell}$ we can deduce from (4.12) that :

$$\forall i \in J_m^{\ell+1} \ , \ \forall j \in J_m^{\ell} \quad u_i \geq u_j \ ;$$

therefore (4.9) is true when we replace ℓ by $\ell+1$.

According to the induction hypothesis (4.9) and to the definition of the algorithm :

$$\begin{aligned} \forall i \in J^{\ell} \quad \phi_i^{\ell} &= \phi_i^{\ell-1} \wedge \left(\bigwedge_{j \in J_m^{\ell}} (k_{i,j} + u_j) \right) \\ &= \left(\phi_i^{o} \wedge \left(\bigwedge_{j \in \mathcal{J}^{\ell-1}} (k_{i,j} + u_j) \right) \right) \wedge \left(\bigwedge_{j \in J_m^{\ell}} (k_{i,j} + u_j) \right) \\ &= \left(\bigwedge_{j \in \mathcal{J}^{\ell}} (k_{i,j} + u_j) \right) \wedge \phi_i \ ; \end{aligned}$$

therefore (4.10) is true with index ℓ , and we can deduce that :

$$(4.13) \qquad \forall \ell \ / \ J^{\ell-1} \neq \emptyset \ , \ \forall i \in J_m^{\ell} \quad u_i = \left(\bigwedge_{j \in \mathcal{J}^{\ell-1}} (k_{i,j} + u_j) \right) \wedge \phi_i$$

and, if we take into account the definition of u^i during the ℓ-th loop, and the nonnegativity of K, we get :

$$(4.14) \qquad \begin{cases} \forall \ell \ / \ J^{\ell-1} \neq \emptyset \\ \forall i \in J_m^{\ell} \quad u_i = \left(\bigwedge_{j \in \mathcal{J}^{\ell}} (k_{i,j} + u_j) \right) \wedge \phi_i \ . \end{cases}$$

Knowing now that (4.9) is true, $\forall \ell \ / \ J^{\ell-1} \neq \emptyset$, we have :

$$\forall i \in \mathcal{J}^{\ell} \ , \ \forall j \in J^{\ell} (J^{\ell} \neq \emptyset) \quad u_i \leq u_j \ .$$

K being non negative, then :

$$(4.15) \qquad \forall i \in \mathcal{J}^{\ell} \ , \ \forall j \in J^{\ell} \quad u_i \leq k_{i,j} + u_j$$

(4.14) and (4.15) give :

$$\forall i \in \{1,\ldots,n\} \quad u_i = \left(\bigwedge_{j \neq i} (k_{i,j} + u_j) \right) \wedge \phi_i$$

Let us suppose that u is not maximal. Let then $v = \{\ldots,v_i,\ldots\}^T$ be another solution of problem (4.4), and be such that $u \leq v$; $u \neq v$.

$\forall i \in J_m^1 \quad v_i > u_i = \phi_i$ is impossible, then

$\forall i \in J_m^1 \quad u_i = v_i$.

Let us suppose that $\forall \ell' < \ell \quad \forall i \in J_m^{\ell'} \quad v_i = u_i$. Then, according to (4.14) :

$$\exists i \in J_m^{\ell} \quad v_i > u_i = \left(\bigwedge_{j \in \mathcal{J}^{\ell}} (k_{i,j} + u_j) \right) \wedge \phi_i$$

is impossible and the proof is achieved.

Remark :

Let $\phi \in R^n$ be such that : $\forall i \quad \phi_i > 0$. Let us suppose that the matrix K verifies :

$$\forall i \in \{1,\ldots,n\} \quad \exists j \neq i \ / \ k_{i,j} = 0$$

Then $u^o = \{0,0,\ldots,0,\ldots,0\}^T$ is solution of the fixed point equation and u^o is a solution distinct from the solution u given by "mono-iteration" relaxation; moreover :

$$u^o < u.$$

"Mono-iteration" relaxation ; form 2 :

Let us consider the following program :

Datas : integer n, vector ϕ, matrix K

"Initialization" $j = \{1,\ldots,n\}$

while $J \neq \emptyset$ do

"loop" begin

$$J_m = \{i_m \in J \quad \forall j_m \in J_m : \phi_{i_m} \leq k_{i_m,j_m} + \phi_{j_m},$$

and $\forall j \in J \ominus J_m \quad \phi_{i_m} < \phi_j\}$.

$J = J \ominus J_m$; $\forall i \in J_m \quad u_i = \phi_i$;

$$\forall i \in J \quad \phi_i = (\underset{j \in J_m}{\wedge} (k_{i,j} + u_j)) \wedge \phi_i ;$$

end;

Proposition 4.2 :

The statement of Proposition 4.1 is left without any change in the case of the above form 2 ; the solution we get there is the same that the corresponding one computed by the form 1 of "mono-iteration" relaxation method.

Proof :

The first part of the statement is proved in an analogous manner to the one of Proposition 4.1, where we modify the implication from (4.13) to (4.14) by taking into account the above new definition of J_m.

In order to prove that the solutions computed respectively in the case of form 1 and form 2 are truly the same, in a first time we consider the situations of a form 1 of "mono-iteration" relaxation, in which J_m would be already reduced to one element of J.

Then let u_{i_ℓ} the component of u computed during the ℓ-th loop. And let v_{i_ℓ} the component, with corresponding index i_ℓ , of the solution got by form 2 during the r-th loop of this algorithm.

Necessarily $u_{i_1} = v_{i_1} = \phi_{i_1} = \min_{i \in \{1,\ldots,n\}} \{\phi_i\}$ and $\{i_1\} = J_m^1 \subset \tilde{J}_m^1$ (where $\tilde{J}_m^1$ is the first occurence of J_m in form 2). We consider the induction hypothesis :

(4.16) $\forall k \leq \ell \quad u_{i_k} = v_{i_k}$

(4.17) $\begin{cases} \text{and if } v_{i_\ell} \text{ is got during the r-th loop of } \underline{\text{form 2}} \text{ of} \\ \text{"mono-iteration" relaxation :} \\ J_m^\ell \subset \tilde{J}_m^r \text{ and } \mathcal{J}^{r-1} \subset \mathcal{J}^\ell \subset \tilde{\mathcal{J}}^r \\ \text{where } \mathcal{J}^r = \bigcup_{r' \leq r} \tilde{J}_m^{r'} \end{cases}$

a) If $i_{\ell+1} \subset \tilde{J}_m^r$, using relation (4.14) which is true either for form 1, or for form 2, we necessarily have :

$$u_{i_{\ell+1}} = v_{i_{\ell+1}} \text{ and } \mathcal{J}^{r-1} \subset \mathcal{J}^{\ell+1} \subset \tilde{\mathcal{J}}^r .$$

b) If $i_{\ell+1} \in \tilde{J}_m^{r+1}$ then by the construction of $J_m^{\ell+1}$ and $\tilde{J}_m^{r+1}$ necessarily : $J_m^{\ell+1} \subset \tilde{J}_m^{r+1}$ and therefore $u_{i_{\ell+1}} = v_{i_{\ell+1}}$ and moreover $\tilde{\mathcal{J}}^r \subset \mathcal{J}^{\ell+1} \subset \tilde{\mathcal{J}}^{r+1}$.

In the cases a) and b), (4.16), (4.17) are true with index $\ell+1$ (with a change of r into r+1 in case b)).

If we consider now situations of form 1 with J_m^ℓ not reduced to one element, we easily verify that it is the same, using the preceding proof, when taking again a form 1 associated to a sequence of $J_m^{\ell'}$, each one beeing reduced to only one element such that :

$$\forall \ell' \quad J_m^{\ell'} \subset J_m^\ell \quad \text{for a right } \ell .$$

4.3. The link between "mono-iteration" relaxation algorithms and the general form (4.8) of relaxations methods ; some consequences.

Proposition 4.3 :

The "mono-iteration" relaxation algorithms can be taken into the general form (4.8) of relaxation method, with initial data $u^o = \phi$. Such "mono-iteration" algorithms are stationary upon a maximal solution after only one iteration.

Proof :

If we take into account the definition of J_m^1 and the property of non negativity of K, we get :

$$\forall i \in J_m^1 \quad u_i = \phi_i = \bigwedge_{j\in\{1,\ldots,m\}} \phi_j = \bigwedge_{j\neq 1} (k_{i,j} + \phi_j) \wedge \phi_i$$

and also :

$$\forall i \in J_m^1 \quad u_i = \Big(\bigwedge_{\substack{j\in J_m^1 \\ j\neq i}} (k_{i,j} + u_j)\Big) \wedge \Big(\bigwedge_{j\in J^1} (k_{i,j} + \phi_j)\Big) \wedge \phi_i$$

Moreover :

$$\forall i \in J^1 \quad \phi_i^1 = \Big(\bigwedge_{j\in J_m^1} (k_{i,j} + u_j)\Big) \wedge \phi_i \leq \phi_i \ ;$$

following (4.10) :

$$\forall \ell \ / \ J^{\ell-1} \neq \emptyset \qquad \forall i \in J^{\ell-1}, \phi_i^{\ell-1} \leq \phi_i$$

by the definition of "mono-iteration" algorithm (form 1 and 2) :

$$\forall i \in J_m^\ell \quad u_i = \phi_i^{\ell-1} = \bigwedge_{j\in J^{\ell-1}} k_{i,j} + \phi_j^{\ell-1} \leq k_{i,j} + \phi_j \ .$$

Therefore, using (4.13), we get :

$$(4.18) \qquad \forall i \in J_m^\ell, u_i = \Big(\bigwedge_{j\in J^{\ell-1}} (k_{i,j} + u_j)\Big) \wedge \Big(\bigwedge_{j\in J^{\ell-1}} (k_{i,j} + \phi_j) \wedge \phi_i\Big)$$

and F being defined by (4.3), we can take relation (4.18) into the equivalent form :

$$(4.19) \qquad \begin{cases} u^0 = \phi \\ \forall i \in J_m^\ell \qquad u_i^\ell = F_i(u_1^{\ell-1}, \ldots, u_j^{\ell-1}, \ldots, u_n^{\ell-1}) \\ \forall i \in J_m^\ell \qquad u_i^\ell = u_i^{\ell-1} \end{cases}$$

which is rightly an algorithm in the form (4.18), which according to

Proposition 4.1 (or Proposition 4.2) is stationary upon the solution u, as soon as $\bigcup_{\ell} J_m^{\ell} = \{1,\ldots,n\}$, that is to say, after one iteration of relaxation (in (4.19), the passage from u^{ℓ} to $u^{\ell-1}$ is corresponding to a "loop" of "mono-iteration" algorithm).

Remark : Let us consider the case where $\forall \ell$ J_m^{ℓ} is reduced to only one element that we take to be j_m. If we perform a permutation P of the unknowns and equations (P being defined by $P(j_m^{\ell}) = \ell$) then, according to the previous proof of Proposition 4.3, we have :

$$\forall \ell \in \{1,\ldots,n\} \quad u_{\ell} = F_{\ell}(u_1,\ldots,u_{\ell-1},\phi_{\ell+1},\ldots,\phi_n)$$

which corresponds then to one iteration of Gauss Seidel's method.

Corollary 4.1 :

If we consider an initial data $u_o \geq \phi$ and a relaxation method of the form of (4.8), where $\forall p$, J_p is reduced to one element, and such that :

$$\forall p \quad \bigcup_{p-n\leq k\leq n} J_k = \{1,\ldots,n\} ,$$

this relaxation method shall be stationnary upon u maximal solution of (4.4), after at most n iterations (in the sense given to the word "iteration" at the beginning of (4.3).

Proof :

At each iteration, we necessarily get a new component u_i of the maximal solution defined by Form 1 of "mono-iteration" algorithm ; the number of such elements being n, this achieves the proof.

Remark : If we take again the statement of corollary, keeping for the sets J_p the general conditions given in the definition of algorithm (4.8) we get the stationnarity of such general algorithms after a finite number of iterations.

4.4. Complexity of Gauss-Seidel's method and "mono-iteration" relaxation method.

Let us suppose that :

(4.20) all the coefficients $k_{i,j}$ of the matrix K, lie in $]0,+\infty[$.

a) The case of Gauss-Seidel's method :

- the evaluation of one component F_i of F needs n comparisons and (n-1) additions.

- each iteration needs n evaluations of components F_i, and then n^2 comparisons and n(n-1) additions.

If we take again the conditions of the statement of corollary 4.1 with an initial data $u^o \geq \phi$, we get stationnarity after, at most,n iterations. Then we can state :

Proposition 4.4 :

With initial data $u^o \geq \phi$ Gauss-Seidel's method needs at most n^3 comparisons and n^2 (n-1) additions.

b) The case of "mono-iteration" relaxation. We consider the form 1 of this method, with J_m^ℓ always reduced to one element.

The execution of the ℓ-th loop needs :

- $(\ell-1)$ comparisons in order to get J_m^ℓ and the ℓ-th component of the solution u.

- $(\ell-1)$ additions and $(\ell-1)$ comparisons in order to get $\phi^{\ell+2}$.

Then we can state :

Proposition 4.5 :

The form 1 of "mono-iteration" relaxation needs, at most,

$n\ (n-1)$ comparisons

$\frac{n\ (n-1)}{2}$ additions.

§ 5 A FAST METHOD FOR THE SOLUTION OF THE BASIC FIXED POINT PROBLEM

5.1 Some comments on the solution of the fixed point problem

The problem (4.4) will be designated in the following paragraphs as P_1. In this chapter, P_1 is analyzed as a dynamic programming problem on a graph and a special fast algorithm is designed to compute the solution. This algorithm (designated A_2 in the following lines) uses a fixed number $n_{op2} = n. \log_2 n$ of operations (additions or comparisons) to compute the solution of P_1. As the mono-iteration algorithm constructed in § 4 (designated A_1 in the following lines) uses a number of operations n_{op1} bounded by $n(n-1)/2$, A_2 can be faster than A_1 for $n \geq 8$.

We describe in this chapter the algorithm A_2 and we give the comparative results obtained in the numerical experimentation of these algorithms.

Remark 1 :

For the problem P_1, it is possible to prove the following property, that enables us to deal with the non-uniqueness problems mentioned in § 4.

Proposition 1 (for the proof see [15]) :

Assume that for all finite sequences of numbers $i_o, i_1, .. i_p (1 \leq i_h \leq n)$

$$k_{i_o i_1} + k_{i_1 i_2} + \ldots + k_{i_p i_o} > 0 \qquad (5.1)$$

then,

there is only one solution u of P_1.

5.2 Structured solution of P_1 : Sequential Dynamic Programming

5.2.1 Introduction of a stepped decision problem

When P_1 derives from the scheduling problem of a set of thermo generators of electrical energy, it has a special structure that makes possible the solution of P_1 with the use of an algorithm of "sequential dynamic programming type".

In the energy problem above mentioned, the parameter "n", is the number of possible states of a system with "q" thermo generator, thus

(5.2) $$n = 2^q .$$

We identify each state with a number "i", in such a form that "i" in its binary notation indicates the state of activity of each thermopower plant, with the following convention :

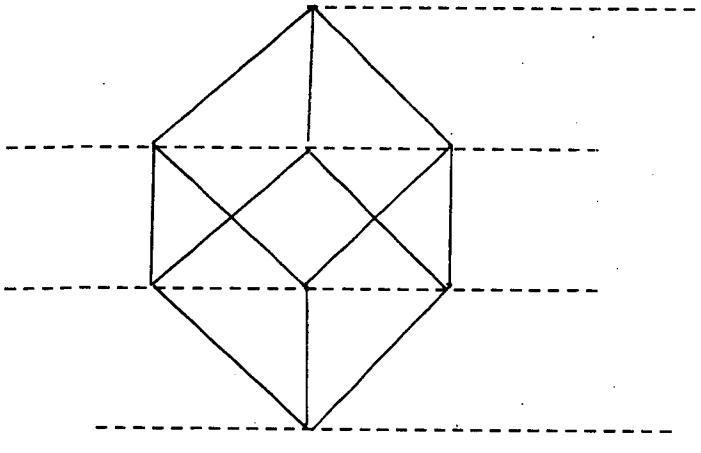
(5.3)
$$\begin{cases} i = \sum_{h=1}^{q} d_h^i \, 2^{h-1} \\ d_h^i = 1 \quad \text{if the } h^{th} \text{ plant is in activity,} \\ d_h^i = 0 \quad \text{if the } h^{th} \text{ plant is inactive.} \end{cases}$$

In this form, we identify the set of "thermic states" with the set of numbers with q -binary digits. Then, we shall use in the following paragraphs, the representation of the state "i" as a vector of q - components :

(5.4) $$i \to d^i = (d_1^i, \ldots, d_q^i)$$

We designate E the set of "thermic states" and we represent it with a graph with (q+1) levels and 2^q points. The level E_h comprises the set of points (or states) that have h-digits with values "1" in their binary notation (h-active plants).

In the graph E, the arrows represent the possible transitions from one level to another (elementary transitions). These transitions take place when there is a start-up of a plant (the state ascends a level) or a shut-down (the state descends a levels).



E_q : 1 state; (1,...1)

E_h : $\binom{q}{h} = q!/(h!(q-h)!)$ states

E_o : 1 state; (o,o,...o)

In the energy problem, the parameters k_{ij} are the cost of transition from the state "i" to the state "j". Then, using the notation (5.3), we have :

$$(5.5) \qquad k_{ij} = \sum_{h=1}^{q} (d_h^j - d_h^i)^+ . r_h^1 + (d_h^j - d_h^i)^- . r_h^o$$

where

r_h^1 is the start-up cost of plant "h"

r_h^o is the shut-down cost of plant "h".

The formula (5.5) introduces a cost for the elementary transitions in the graph E, and by concatenation of elementary transitions, for any arbitrary transition.

For each initial state i_o and to any path

$$\gamma_{i_o} = (i_o, i_1, \ldots, i_p) \ , \ i_h \in \{o, 1, \ldots, 2^q - 1\}$$

being γ_{i_o} a concatenation of elementary transitions, we assign the cost $J(i_o, \gamma_{i_o})$

$$(5.6) \qquad J(i_o, \gamma_{i_o}) \begin{cases} = k_{i_o i_1} + \ldots k_{i_{p-1} i_p} + \phi_{i_p} & \text{if } p \geq 1 \\ = \phi_{i_o} & \text{if } p = o \end{cases}$$

We define G_{i_o} as the set of all possible paths γ_{i_o} with origin i_o, and we introduce the following problem :

P_2 : find the function $V(i_o)$, where

$$(5.7) \qquad V(i_o) = \inf_{\gamma_{i_o} \in G_{i_o}} J(i_o, \gamma_{i_o})$$

With the usual techniques of dynamic programming theory, it is possible to prove the following property :

Theorem 5.1 :

Assuming (5.1) satisfied, the function V(i) is well defined and bounded moreover, V is equal to the unique solution u of P_1 and in consequence, P_1 and P_2 are equivalent. □

By virtue of this equivalence, we can solve P_1 applying to P_2 the algorithms given by dynamic programming theory. To develop an efficient algorithm, we shall use the following result :

Theorem 5.2 :

The optimum of P_2 is realized by a path which lenght is $p \leq q$ □

As a consequence, P_2 is equivalent to the following problem P_3, obtained by restriction of P_2 to the set of paths with lenght $p \leq q$.

P_3 : Find the function $V(i_o)$ defined by :

$$(5.8)\quad V(i_o) = \min_{\gamma_{i_o} \in G'_{i_o}} J(i_o, \gamma_{i_o})$$

where

$$(5.9)\quad G'_{i_o} = \{\gamma_{i_o} = (i_o, i_1, \ldots i_p) / \; p \leq q \;,\; i_h \in \Gamma(i_{h-1}) \;,\quad \forall h = 1,p\}$$

where

(5.10) $\Gamma(i)$ = {j/j has at most one binary digit different from those of i}.

5.2.2 Recursive solution of P_3

To solve P_3, we introduce an auxiliar problem P_4.

P_4: Find the function $V^-(i_o)$ defined by

$$(5.11)\quad V^-(i_o) = \min_{\gamma_{i_o} \in D_{i_o}} J(i_o, \gamma_{i_o})$$

where

$$(5.12)\quad D_{i_o} = \{\gamma_{i_o} = (i_o, i_1, .., i_p) /\ i_h \in \Gamma^-(i_{h-1}),\ h = 1,p \quad p \leq q\}$$

and

$$(5.13)\quad \Gamma_o^-(i) = \{j/j \text{ has at most one more binary digit with value "o" than i}\}$$

i.e., D_{i_o} is the set of non-ascending paths and $V^-(i_o)$ is the optimum of J in this set of paths.

In the following lines, we give a recursive formula that computes the function $V(i_o)$, using the previously computed values of $V^-(.)$.

For any path, as can be seen from (5.5) and (5.6), the cost J depends only on the digits that change its values and is independent of the order of the changes ; then, any path can be replaced by an equivalent one such that the first part is ascending and the final part is descending.

In consequence, we consider only the paths of the following form :

$$(5.14)\quad \left|\begin{array}{l} \gamma_{i_o} = (i_o, i_1, \ldots, i_r, j_1, \ldots, j_s) \\ s+r \leq q \\ i_h \in \Gamma^+(i_{h-1}) \quad \forall h = 1,r \\ j_h \in \Gamma_o^-(j_{h-1}) \quad \forall h = 1,s \\ \text{(we define } j_o = i_r) \end{array}\right.$$

where

$$(5.15)\quad \Gamma^+(i) = \{j/j \text{ has only one binary digit different from those of i and that digit takes the value "1"}\}$$

$$(5.16)\quad \Gamma^-(i) = \{j/j \text{ has only one binary digit different from those of i and that digit takes the value "o"}\}$$

$$(5.17)\quad \Gamma_o^-(i) = \Gamma^-(i) \cup \{i\}$$

We define now two subsets of paths :

(5.18) $D_{i_o} = \{\gamma_{i_o} / r = o\}$

(5.19) $G^+_{i_o} = \{\gamma_{i_o} / r > o\}$

Then, G'_{i_o} is equivalent to $D_{i_o} \cup G^+_{i_o}$ and in consequence :

$$(5.19)\quad V(i_o) = \min_{\gamma_{i_o} \in G'_{i_o}} J(i_o, \gamma_{i_o}) = \min_{\gamma_{i_o} \in D_{i_o} \cup G^+_{i_o}} J(i_o, \gamma_{i_o}) =$$

$$= \min_{\gamma_{i_o} \in G^+_{i_o}} J(i_o, \gamma_{i_o}) \wedge \min_{\gamma_{i_o} \in D_{i_o}} J(i_o, \gamma_{i_o})$$

D_{i_o} is the set of non-ascending paths then we have, by virtue of (5.11)

$$(5.20)\quad V(i_o) = (\min_{\gamma_{i_o} \in G^+_{i_o}} J(i_o, \gamma_{i_o})) \wedge V^-(i_o).$$

To compute the minimum of J in $G^+_{i_o}$, we apply the dynamic programming technique and we obtain the recursive relation :

$$(5.21)\quad \min_{\gamma_{i_o} \in G^+_{i_o}} J(i_o, \gamma_{i_o}) = \min_{i_1 \in \Gamma^+(i_o)} (k_{i_o i_1} + V(i_1)).$$

Then, from (5.20)-(5.22) we obtain

$$(5.22)\quad V(i_o) = (\min_{i_1 \in \Gamma^+(i_o)} (k_{i_o i_1} + V(i_1))) \wedge V^-(i_o)$$

Remark 2 :

if $i_o = 2^q - 1$, it is $\Gamma^+(i_o) = \emptyset$ and then, from (5.23) we have the relation :

$$V(2^q-1) = V^-(2^q-1) \tag{5.24}$$

This relation gives the initial value necessary to start the descending recurrence (5.23) on the graph E, i.e., from (5.23), (5,24) we can compute the values of $V(i_o)$ $\forall\, i_o \in E_{q-1}$, then the values of $V(i_o)$ $\forall i_o \in E_{q-2}$, etc., until we reach E_o.

To complete the developement of algorithm A_2, we must obtain a recursive formula to compute $V^-(.)$. In problem P_4, there are two types of possible decisions :

a) the path γ_{i_o} has null length, then

$$J(i_o, \gamma_{i_o}) = \phi_{i_o} \tag{5.25}$$

b) the path γ_{i_o} descends at least one step ; then, applying the dynamic programming principle, we obtain for the minimum with this type of policies, the following value :

$$\min_{i_1 \in \Gamma^-(i_o)} (k_{i_o i_1} + V^-(i_1)) \tag{5.26}$$

$V^-(i_o)$ is the minimum of the values a, b; then we obtain the following (ascending) recurrence :

$$V^-(i_o) = \left(\min_{i_1 \in \Gamma^-(i_o)} (k_{i_o i_1} + V^-(i_1)) \right) \wedge \phi_{i_o} \tag{5.27}$$

Remark 3

If $i_o = o$, it is $\Gamma^-(o) = \emptyset$; then, from (5.27) we have

$$V^-(o) = \phi_o \tag{5.28}$$

This value is the starting value for the recurrence (5.27) i.e., from (5.27),(5.28) we can obtain all the values $V^-(i_o)$, $\forall i_o \in E_1$, then $V^-(i_o)$ $\forall i_o \in E_2$, etc.

5.2.3 The recursive algorithms A_2

step 0 : set $V^-(o) = \phi(o)$, $h = 1$

step 1 : $\forall i \in E_h$,

$$\text{set } V^-(i) = \Big(\bigwedge_{j \in \bar{\Gamma}(i)} (k_{ij} + V^-(j)) \Big) \wedge \phi_i$$

step 2 : If $h = q$, go to step 3 ; else

set $h = h+1$ and go to step 1.

step 3 : set $V(2^q-1) = V^-(2^q-1)$, $h = q-1$

step 4 : $\forall i \in E_h$, set

$$V(i) = \Big(\bigwedge_{j \in \Gamma^+(i)} k_{ij} + V(j) \Big) \wedge V^-(i)$$

step 5 : if $h = o$, stop ; else, set $h = h-1$ and go to step 4

5.3 Complexity of algorithm A_2

The complexity of algorithm A_2 is measured by the number of additions or comparations employed in the computation.

These operations are performed in steps 1 and 4.

In step 1, the total of additions is :

$$n_a^1 = \sum_{h=1}^{q} \binom{q}{h} . h$$

and the number of comparisons is the same :

$$n_c^1 = \sum_{h=1}^{q} \binom{q}{h} . h = n_a^1$$

In step 4, a number of n_a^4 additions and n_c^4 comparisons are performed, where

$$n_a^4 = n_c^4 = \sum_{h=o}^{q-1} \binom{q}{h} (q-h)$$

Then, the total number of additions and comparisons are :

$$n_a = n_a^1 + n_a^4 = \sum_{h=o}^{q} \binom{q}{h} . \; q = 2^q . q$$

$$n_c = n_c^1 + n_c^4 = q . 2^q$$

From (5.2), we obtain that the complexity of algorithm A_2 is given by

(5.29) $$n_{op2} = n . \log_2 n$$

As the complexity of algorithm A_1 is given by $n_{op1} = \frac{1}{2} n(n-1)$, A_2 is faster than A_1 for $n \geq 8$ (corresponding to $q \geq 3$), as it is shown by the following table.

q	n	n(n-1)/2	$n . \log_2 n$
1	2	1	2
2	4	6	8
3	8	28	24

5.4 Number of operations and computing times of algorithms A_1 and A_2

We have done the numerical experimentation of both algorithm A_1 and A_2. The computer program (written in Basic language), makes possible the comparation of the number of operations and the times of computation employed by each algorithm. In the following table, we show the values of times and number of operations obtained for different data $(n, \phi, r^{\circ}_h, r^1_h)$. The values are given for the two computers employed (TI 99 and CII-Honeywell-Bull/DPS 68)

q	n	n_{op1}	n_{op2}	TI 99 T_1	TI 99 T_2	CII-HB T_1	CII-HB T_2
3	8	27	24				
4	16	120	64	1'03"	14"	0.05"	0.01"
4	16	93	64	45"	14"		
4	16	119	64	1'02"	14"		
5	32	491	160	4'30"	37"	0.26"	0.02"
5	32	386	160	3'17"	37"		
6	64	1838	384	18'22"	1'26"	1.24"	0.05"
6	64	1491	384	15'30"	1'26"		
6	64	1686	384	16'44"	1'26"		
7	128	6909	896	1h 10'	3'10"		
7	128	8128	896	1h 30'	3'10"	5.79"	0.11"
8	256	32641	2048		7'02"	26.13"	0.25"
9	512	130817	4608			1'56"61	0.67"

TABLE 1

The algorithm A_1 (analyzed in § 4), solves the fixed point problems P_1 efficiently, in a form that is independent of the choice of the starting point for the iteration (the algorithm presented in [9] is strongly dependent of that choice), and it obtains the solution in a finite number of operations that use a number of operations $n_{op1} \leq n(n-1)/2$. In the energy problem that is the objective of this paper, $n=2^q$ and in consequence, n_{op1} takes greats values as q grows. The algorithm A_2 (of dynamic programming type) presented in this chapter, also solves P_1 in a fixed number of steps using a number of operations $n_{op2} = n. \log._2 n$ and it is faster than A_1 for $q \geq 3$ ($n \geq 8$) as is shown in the above table of computing times.

As a final information we give the times needed (on a HB 68) using the graph approach for a whole run of the algorithm during one period of the time interval, with the hydropower stock interval including 19 points of discretization while the thermo power ranges were covered by 5 points :

Number of thermo power plants	2	3	4
Time	1''32	11''	2'09''

TABLE 2

So, rather simple systems can be optimized in real time using this procedure.

5.5 More complex systems

When considering systems with more than one dam we have multidimensional state vectors x in our QVI equations. Discretization schemes are now more complicated and computer times increase significately each time we add a thermo power plant (Table 3).

Number of dams	2	2	2	2
Number of thermo power plants	2	3	4	5
Time	25''	3'00''	21'00''	$3^h36'$

TABLE 3

Nevertheless, using a more efficient mininization over the power range when solving the QVI'S corresponding to two dams we obtain the times given in table 4.

Time of HB 68 computer correspond to one period of the time interval when the hydropower ranges and the thermo power ranges were covered by 19 and 5 points respectively.

Number of dams	2	2	2	2
Number of thermo power plants	2	3	4	5
Time	0'5"	0'37"	4'04"	25'00"

TABLE 4

The two following tables give the "sensibility" of the algorithm to variations in the number of discretization points.

Times are always for one period of the time interval. The models considered had two dams.

In table 5 the hydropower range is always covered by 19 points.

In table 6 we retain 4 thermo power discretization points.

Number of thermo power discretization points / Number of thermo power plants	5	4	3
2	0'5"	4"	3"
3	0'37	24"	15"
4	4'04	2'12"	1'08"
5	25'	12'47"	5'17"

TABLE 5

Number of hydropower discretization points	19	15	11	5
Number of thermo power plants : 4	2'12"	1'22"	44"	13"

TABLE 6

Simulation routines have been written. When typing on a terminal the initial values (i.e. : starting period, number of periods to be simulated, hydraulic stock of different dams, number of thermal plants already working and their identification) the routine prints the desired optimal trajectory to the last period by searching the adequate values in a file previously created during the optimization run. It gives the future state of the system at each period together with the optimal production levels.

5.6 Example 1

The demand is given in table A. Each day is divided in 3 periods. The duration (in hours) of each period is shown in the table.

DEMAND

DAY	MONDAY			TUESDAY			WEDNESDAY			THURSDAY			FRIDAY			SATURDAY			SUNDAY		
PERIOD	1	2	3	1	2	3	1	2	3	1	2	3	1	2	3	1	2	3	1	2	3
Duration (in hours)	10	10	4	8	10	6	8	10	6	8	10	6	8	10	6	4	11	9	4	10	10
Demande (en GW/h)	84.1	78.5	73.0	87.6	83.0	79.5	88.8	84.3	80.9	92.1	87.6	84.3	85.3	80.9	77.9	77.2	68.4	62.8	82.7	75.4	67.8

TABLE A

The energy production systems consists in a dam, sveral thermo power plants and an additional source which is available if it is required.

The characteristics of the dam are the following :

Maximum capacity of storage	:	7322	GWh
Maximum power (in turbine)	:	6.58	GW
Imputs	:	1.06	GWh
Initial stock	:	287.8	GWh

The cost of KWh in stock is given in c/KWh in table B

DAY	MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY	SATURDAY	SUNDAY
Stock between 244.1 and 7322 Gwh	78	74	70	68	65	66	67
Stock between 0 and 244.1 Gwh	160	155	150	148	144	144	145

TABLE B

The admissible level of production in GW, the unit cost in G/KWh and the number of hours of heating (at maximum admissible power) each plant before a start-up are given in TABLE C for the remaining plants :

THERMIC CHARACTERISTICS

DAY	NUCLEAR	COAL 1	COAL 2	FUEL 1	FUEL 2	GAS	ADDITIONAL SOURCE
1 Monday	64,24	5,37	3,62	3,86	1,84	2,11	∞
2 Tuesday	64,61	6,24	3,92	3,65	1,12	2,23	∞
3 Wednesday	66,14	7,70	4,48	4,06	1,66	2,11	∞
4 Thursday	67,09	5,76	3,88	4,12	1,72	2,37	∞
5 Friday	64,27	7,07	3,87	3,44	1,29	2,00	∞
6 Saturday	65,41	7,03	4,80	3,68	0,92	1,86	∞
7 Sunday	66,37	5,57	3,55	3,29	0,85	2,10	∞
Unit cost	5,15	22,0	22,2	50,1	56,5	111,0	200,
Duration of heating (in hours) before a start-up	-	6	6	6	6	-	-

TABLE C

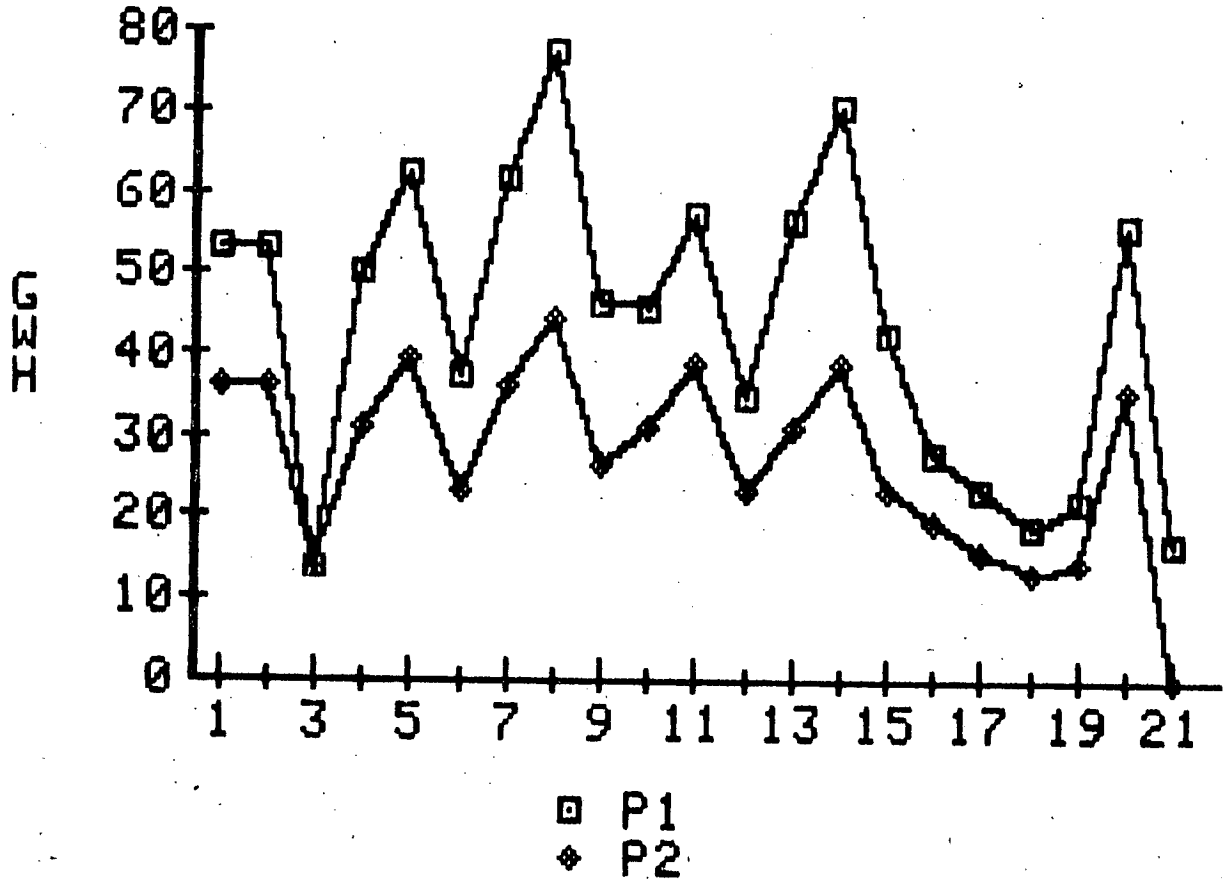
The optimization was done using the graph procedure. For the thermo power plant we consider 4 discretization points :

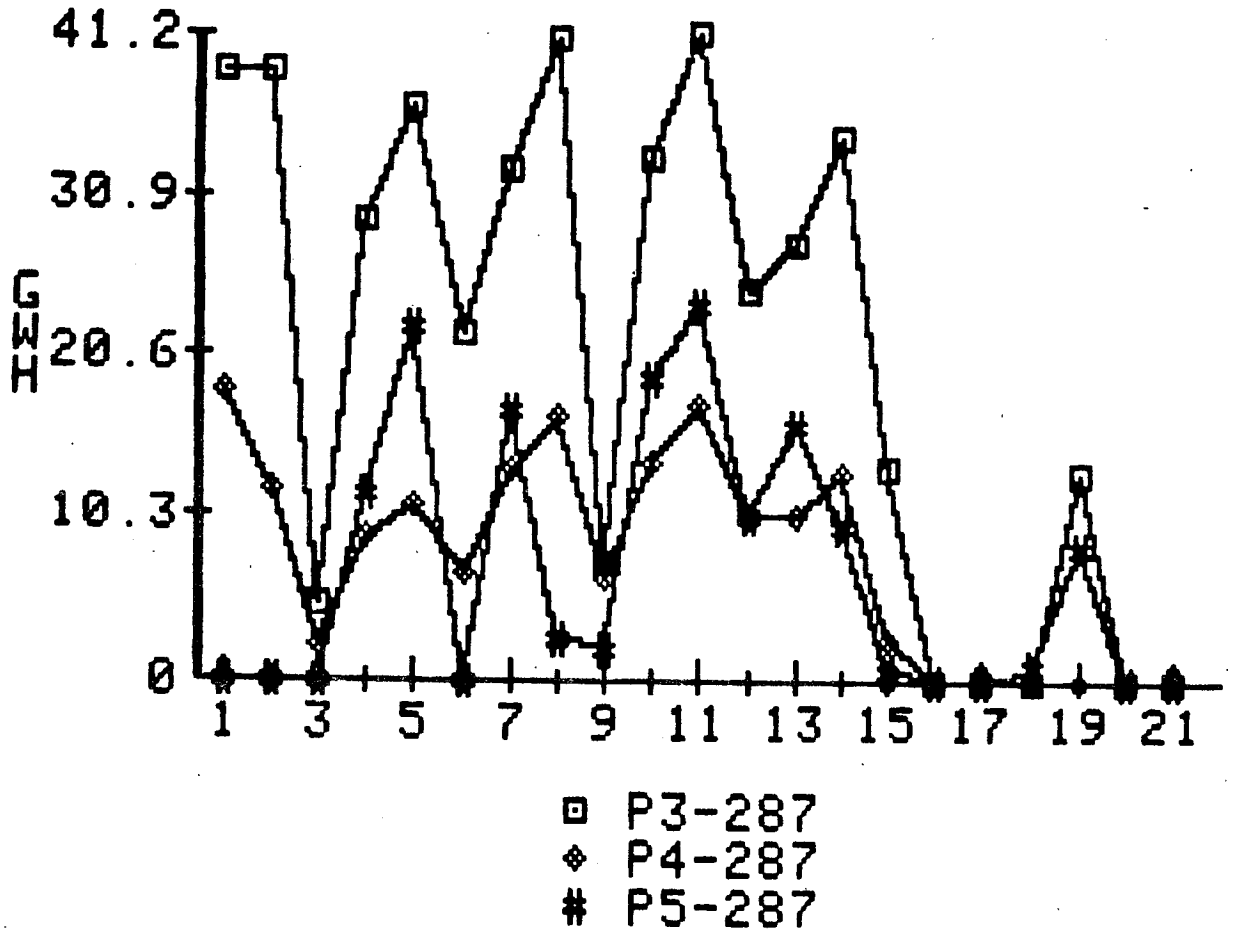
$$0 \quad ; \quad P_{min} = 0,25 \; P_{max} \quad ; \quad 1/2 \; (P_{min} + P_{max}) \; ; \; P_{max}$$

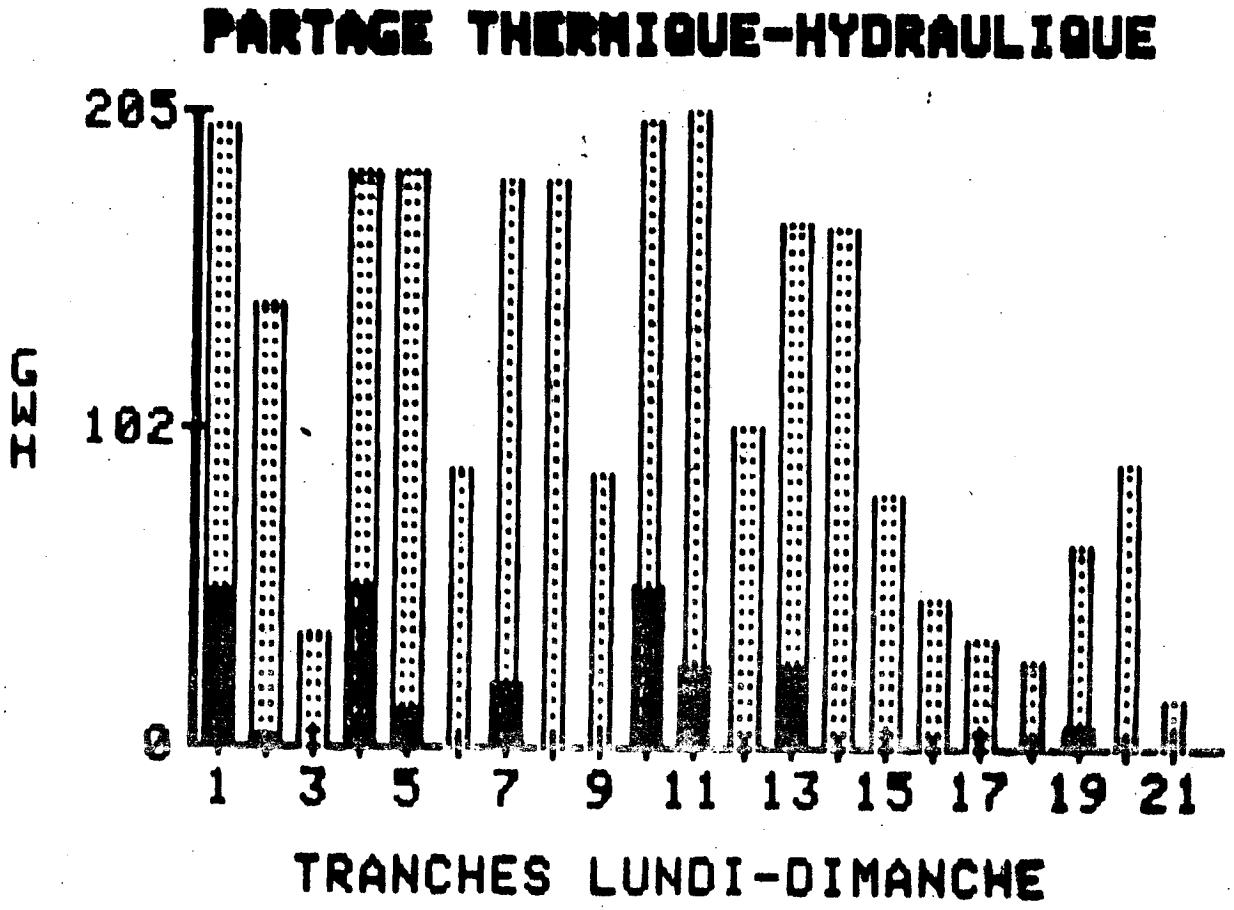
while the hydropower stock was divided in two intervals.

The time of computation for the whole optimization (the week = 21 periods) was 7''54 on a HB 68 DPS/Multics Computer.

The optimal production policy is shown in the following figures. Nuclear production is omitted (it is always used as the first possibility). The production of COAL 1 is denoted by P_1 ; COAL 2 by P_2 ; FUEL 1 by P_3 ; FUEL 2 by P_4 and GAS by P_5.


GEH
80
70
60
50
40
30
20
10
0
1
3
5
7
9
11
13
15
17
19
21
P1
P2


41.2
30.9
20.6
10.3
0
GWH
1
3
5
7
9
11
13
15
17
19
21
P3-287
P4-287
P5-287


PARTAGE THERMIQUE-HYDRAULIQUE
205
102
0
GWH
1 3 5 7 9 11 13 15 17 19 21
TRANCHES LUNDI-DIMANCHE

EXAMPLE 2

We introduce in the production system of example 1 a second dam with the following characteristics :

Max. Capacity : 361 GWh ; Max. level of production : 2.24 GW

Imputs : 0.01 GW/h ; Initial stock : 88.3 GWh

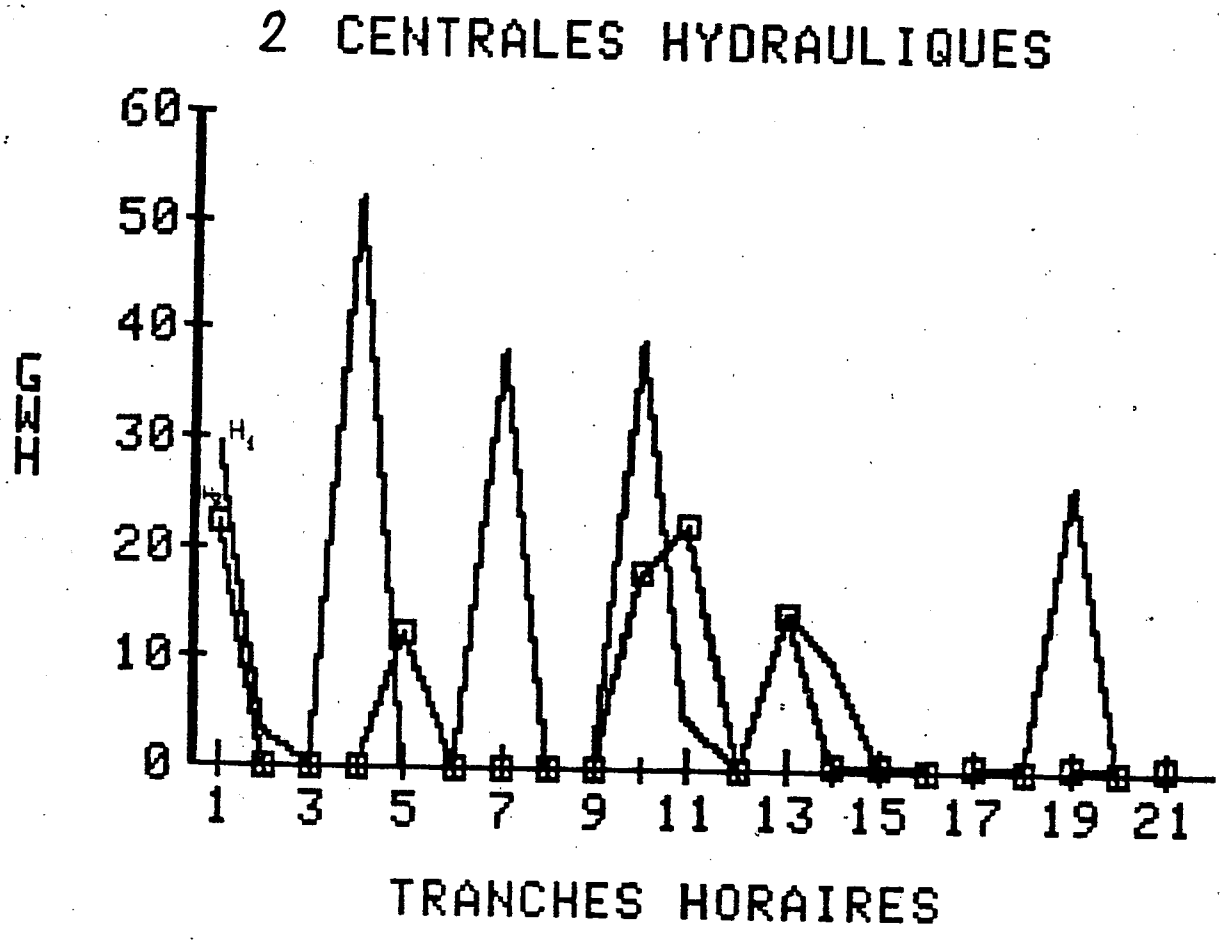
The cost of production, in c/KWh is given in Table D

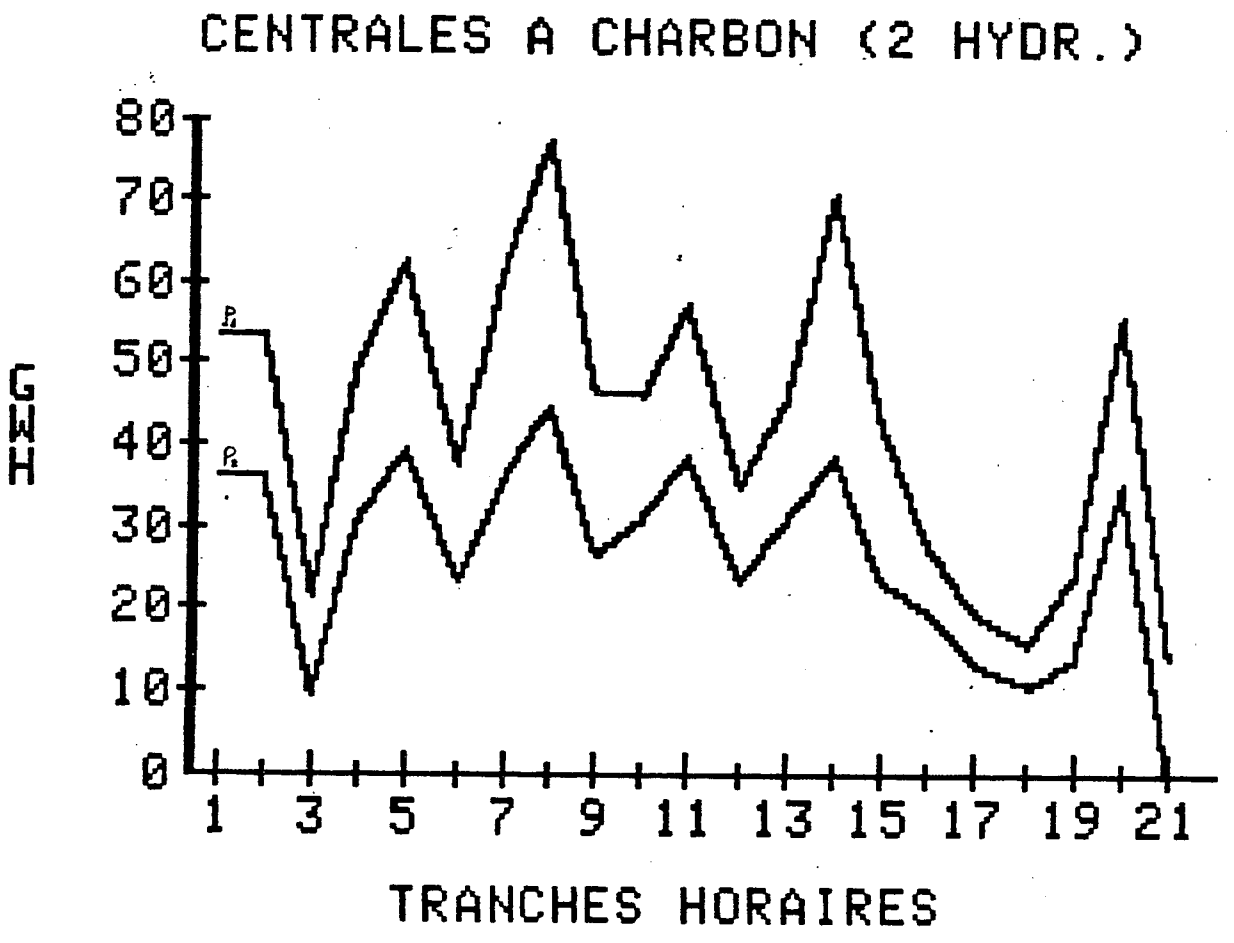
DAY	MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY	SATURDAY	SUNDAY
Stock between 72.2 and 361 GWh	74	70	65	63	59	60	61
Stock between 0 and 72.2 GWh	145	140	130	120	92	100	110

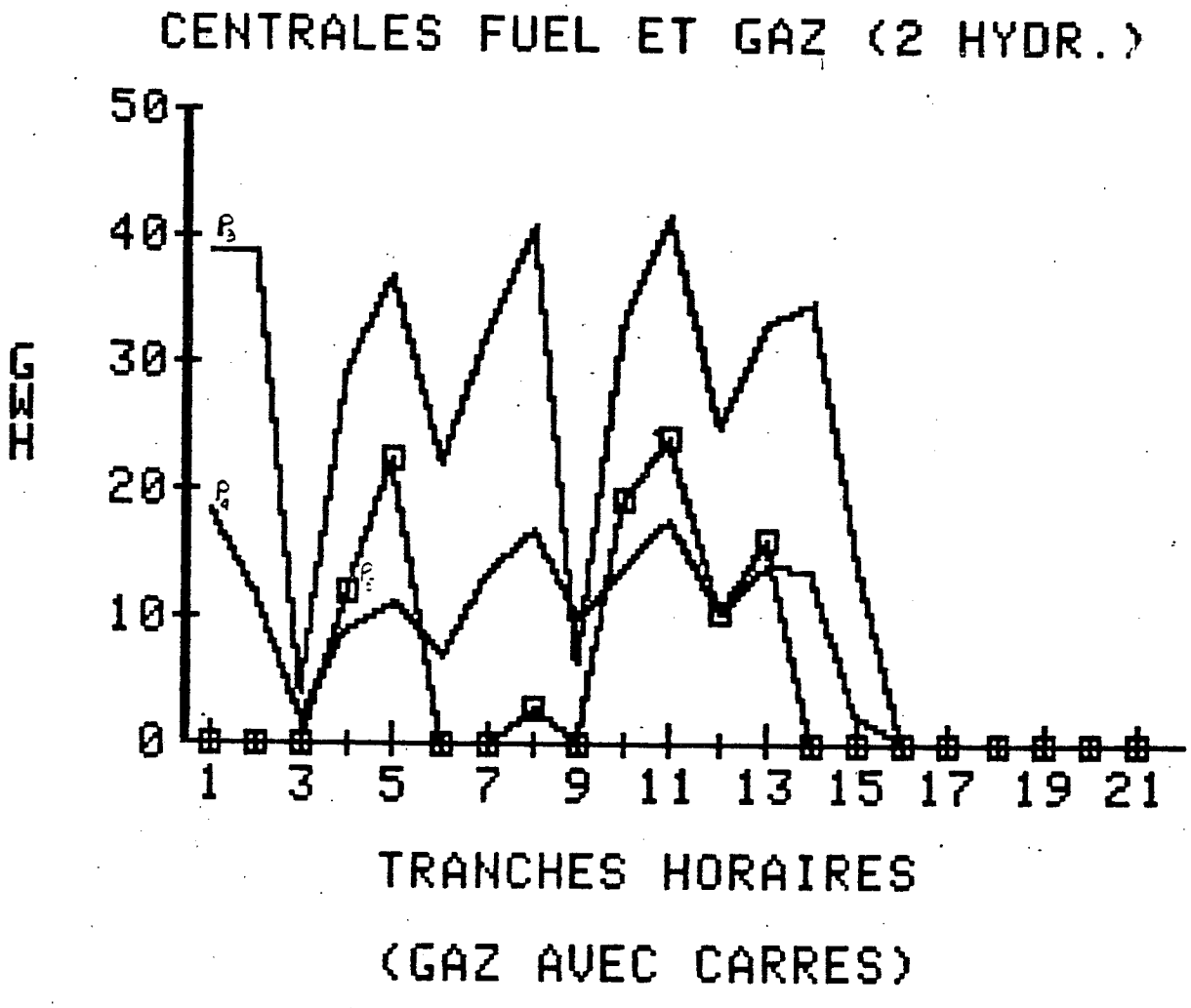
TABLE D

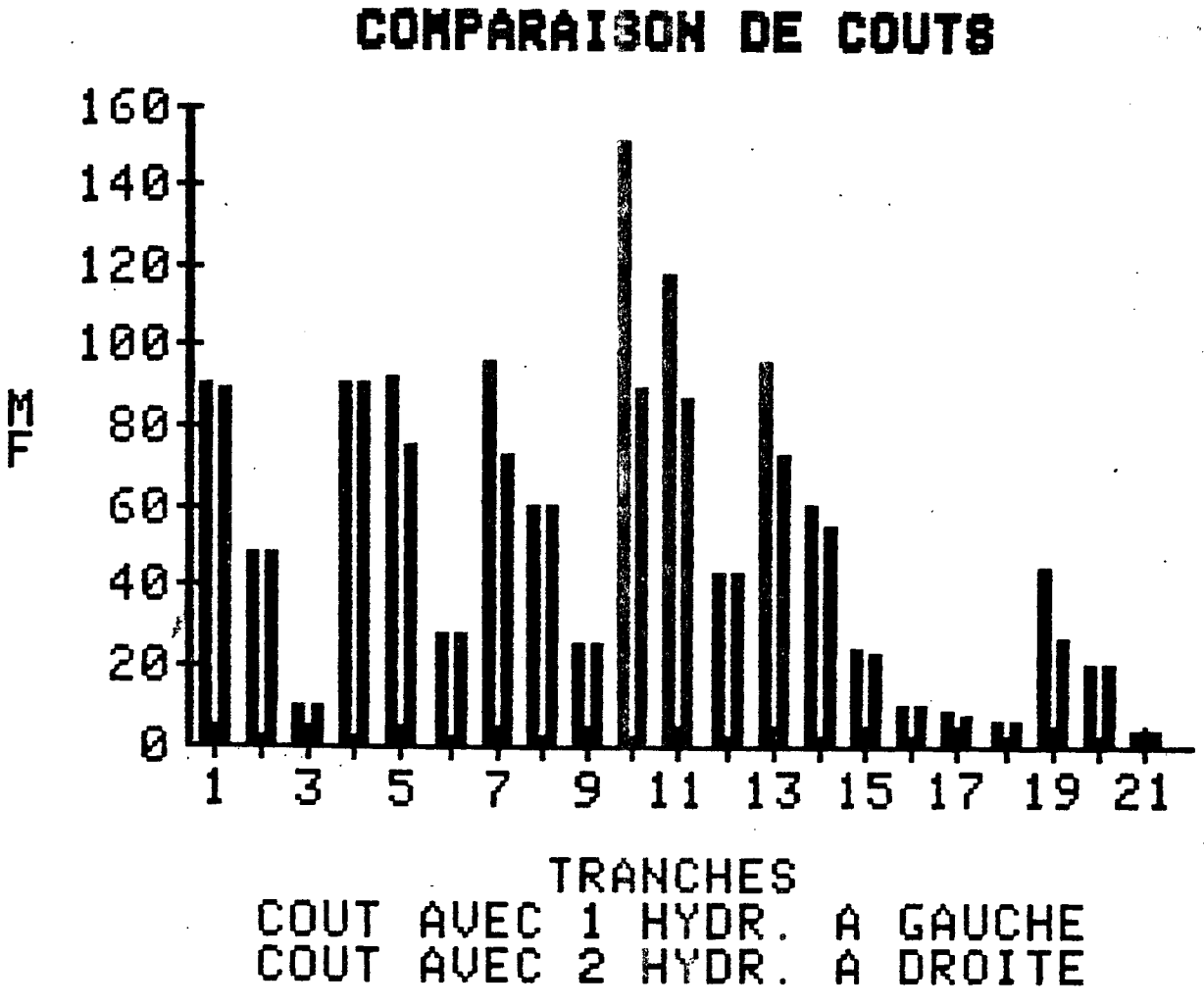
In this case the optimization was done (always in a HB 68) in 19''.

After simulation, the optimal policy is given in the following figures :


2 CENTRALES HYDRAULIQUES
60
50
40
30
20
10
0
1
3
5
7
9
11
13
15
17
19
21
TRANCHES HORAIRES


CENTRALES A CHARBON (2 HYDR.)
GWH
80
70
60
50
40
30
20
10
0
1
3
5
7
9
11
13
15
17
19
21
TRANCHES HORAIRES


CENTRALES FUEL ET GAZ (2 HYDR.)
GWH
50
40
30
20
10
0
1
3
5
7
9
11
13
15
17
19
21
TRANCHES HORAIRES
(GAZ AVEC CARRES)


COMPARAISON DE COUTS
160
140
120
100
80
60
40
20
0
MF
1
3
5
7
9
11
13
15
17
19
21
TRANCHES
COUT AVEC 1 HYDR. A GAUCHE
COUT AVEC 2 HYDR. A DROITE

BIBLIOGRAPHY

[1] M.C. BANCORA-IMBERT, R. GONZALEZ, J.C. MIELLOU, E. ROFMAN : "An algorithm to optimize energy-production systems in real time" Int. Journal of Energy systems (IASTED), Vol. 4, 1984.

[2] A. BENSOUSSAN, J.L. LIONS : "Contrôle impulsionnel et inequations quasi variationnelles", Dunod, Paris 1982.

[3] M.N. EL TARAZI : "Thèse de Doctorat ès Sciences - Besançon 1981

[4] M.N. EL TARAZI : "Sur les algorithmes mixtes par blocs de type Newton-relaxation" C.R.A.S. Paris, 283 Serie A, p. 721-724 (1976)

[5] M. GONDRAN - M. MINOUX : " Graphes et algorithmes" Collection de la direction des études et recherches d'E.D.F. n° 37, Eyrolles 1979.

[6] R. GONZALEZ : "Some contributions to the optimisation of energy production systems" Working paper, Conicet - Univ. Nac. Rosario.

[7] R. GONZALEZ : "Sur la résolution de l'éq. de H. Jacobi du Contrôle Deterministe" Cahiers de Math. de la Décision, CEREMADE, n° 8029, 8029 b, 1980.

[8] R. GONZALEZ, E. ROFMAN : "Problèmes de control bang-bang con intervales de tiempo minimo entre commutaciones" in Recents methods in non linear analysis" Ed. De Giorgi-Magenes-Mosco, Pitagora Ed. Bologne 1978.

[9] R. GONZALEZ, E. ROFMAN "On deterministic control problems : an approximation procedure for the optimal cost" part I and II. SIAM J. on Control and Optimization, to appear. See also RR 151, INRIA, Rocquencourt, France.

[10] C. LEGUAY : "Application du contrôle stochastique à un problème de gestion optimale d'énergie". Thèse de Docteur-Ingénieur, Université de Paris IX, 1975.

[11] J.C. MIELLOU : "A mixed relaxation algorithm applied to QVI. Lectures notes in Comp.Sc. 41 Spinger(1975) p. 192-199.

[12] J.C. MIELLOU : "Iterations chaotiques à retards. Etude de la convergence dans le cas d'espaces partiellement ordonnés". CRAS. Paris 280, Série A, p.233, (1975).

[13] NAKESHBENDI : "Quelques problèmes numériques et non numériques en calcul parallèle" Thèse 3ème cycle, Besançon 1979.

[14] J.M. ORTEGA, J.C. RHEINBOLDT : "Iterative solution of non linear equations in several variables" Academic Press 1970

[15] E. ROFMAN : "Approximation of Hamilton-Jacobi-Bellman equations in deterministic control theory. An application to energy production systems" in Recents Mathematical Methods in Dynamics Programming, Springer, Lecture Notes in Mathematics, Rome, 1984, to appear.

Imprimé en France

par

l'Institut National de Recherche en Informatique et en Automatique